Claims

1. (Original) A method comprising:

identifying state information comprising a transfer from a first simulation model in a simulation environment, said transfer being directed to a second simulation model in a circuit design being simulated in the simulation environment;

receiving the state information from the first simulation model; and

making the state information available to the second simulation model without simulating the transfer in the circuit design.

2. (Original) The method of claim 1 wherein simulating the transfer from the first simulation model to the second simulation model in the circuit design comprises transferring the state information through at least one additional simulation model in the simulation environment.

3. (Original) The method of claim 1 wherein receiving the state information and making the state information available comprises:

storing the state information in a coherent state memory space that is part of the simulation environment and corresponds to an element in the circuit design being simulated, said coherent state of memory space being accessible to both the first simulation model and the second simulation model.

4. (Original) The method of claim 3 wherein the coherent state memory space is accessible to a plurality of additional simulation models.

5. (Original) The method of claim 1 wherein receiving the state information and making the state information available comprises at least one of:

a virtual transfer path for use when a simulation model of a transfer path in the circuit design is not included in the simulation environment; and

a higher performance transfer path than the simulation model of the transfer path in the circuit design.

6. (Currently amended) The method of claim 5 wherein the higher performance ~~performace~~ transfer path provides a lower level of resolution than the simulation model of the transfer path in the circuit design.

7. (Original) The method of claim 3 wherein the simulation environment comprises a plurality of additional simulation models, each of the plurality of additional simulation models corresponding to one or more of a plurality of additional coherent state memory spaces, the method further comprising:

identifying additional state information comprising additional transfers among the plurality of additional simulation models in the simulation environment; and

storing the additional state information in appropriate ones of the plurality of additional coherent state memory spaces such that the additional state information is accessible to corresponding ones of the plurality of additional simulation models without simulating the additional transfers in the circuit design.

8. (Original) The method of claim 1 wherein the simulation environment comprises a plurality of simulation domains, the method comprising:

selectively activating and deactivating particular simulation domains in the simulation environment such that a resolution and a performance for the circuit design being simulated is dynamically modified as the state information is received and made available.

9. (Original) The method of claim 1 wherein the plurality of simulation domains comprise at least one of a software execution domain, a hardware simulation domain, and an abstract model simulation domain.

10. (Currently amended) The method of claim 9 wherein the software execution domain comprises at least one ~~on e~~ of a native processor package, an instruction set simulator (ISS), and a programming language simulator to model software execution in one or more processors.

11. (Original) The method of claim 9 wherein the hardware simulation domain comprises at least one or a logic simulator and a programming language simulator.

12. (Original) The method of claim 11 wherein the logic simulator comprises one of a hardware description language (HDL) based simulator, a gate-level simulator, a simulation accelerator, a system simulator, a cycle simulator, and a programmable hardware emulator.

13. (Original) The method of claim 11 wherein the programming language simulator comprises at lease one of a C programming language simulator, a C++ programming language simulator, a simulator using a C-based language, a simulator using a C++ based language, and a JAVA programming language simulator.

14. (Original) The method of claim 9 wherein the hardware simulation domain comprises at least one simulation model of a circuit element in the circuit design.

15. (Original) The method of claim 8 further comprising:

partitioning the circuit design into the plurality of simulation domains based on a partition criteria.

16. (Original) The method of claim 15 wherein the partition criteria comprises at least one of an abstraction level, a simulation type, and a function type.

17. (Original) The method of claim 16 wherein partitioning the circuit design based on the abstraction level partitions the circuit design into at least one of a pin-level domain, a bus-level domain, and a transaction-level domain.

18. (Original) The method of claim 16 wherein partitioning the circuit design based on the simulation type partitions the circuit design into at least one of a software execution domain, a logic simulator domain, and a programming language simulator domain.

19. (Original) The method of claim 16 wherein partitioning the circuit design based on the function type comprises:

identifying one or more functional elements in the circuit design that have a particular level of independent operation from the remainder of the circuit design; and

defining a domain encompassing each identified functional element.

20. (Original) The method of claim 8 wherein each of the plurality of simulation domains provides a particular performance level and a particular resolution level, and wherein the particular simulation domains are selectively activated or deactivated during particular stages of simulation in combinations that either accelerate performance of the simulation environment or increase resolution of the simulation environment.

21. (Original) The method of claim 8 wherein selectively activating and deactivating the particular simulation domains comprises:

identifying a system state of the circuit design;

determining which of the plurality of simulation domains are to be active for the identified system state; and

advancing simulation time only in each activated simulation domain.

22. (Original) The method of claim 21 wherein determining which of the plurality of simulation domains are to be active for the identified system state comprises at least one of a centralized control, a transaction-based control, and a distribution control.

23. (Canceled)

24. (Original) The method of claim 22 wherein the system state comprises system addresses in the circuit design.

25. (Original) The method of claim 22 wherein the system state comprises a data transaction in the circuit design, said data transaction being configured with information identifying which of the plurality of simulation domains are to be active for the data transaction, and wherein the transaction-based control comprises:

sending a message to a centralized simulation clock as part of the data transaction, said message to instruct the centralized simulation clock with respect to which of the plurality of simulation domains are to be active for the data transaction.

26. (Original) The method of claim 22 wherein a predetermined simulation domain is configured with activation information identifying at least one particular system state for which the predetermined simulation domain is to be active, wherein identifying the system state comprises receiving a broadcast of the system state at the predetermined simulation domain, and wherein distributed control at the predetermined simulation domain comprises:

determining if the predetermined simulation domain is to be active for the identified system state based on the activation information; and

advancing an operation in the predetermined simulation domain accordingly.

27. (Original) The method of claim 26 wherein the information further identifies an event for terminating operation of the predetermined simulation domain for the at least one particular system state.

28. (Original) The method of claim 21 wherein determining which of the plurality of simulation domains are to be active for the identified system state depends on a plurality of control mechanisms, wherein each of the plurality of control mechanisms comprises a priority level, and wherein a higher priority control mechanism takes precedence over a lower priority control mechanism.

29. (Original) The method of claim 8 wherein the plurality of simulation domains comprise a hierarchical structure, and wherein selectively activating and deactivating the particular simulation domains is based on levels of the hierarchical structure.

30. (Original) The method of claim 1 wherein both the first simulation model and the second simulation model are within a same simulation domain in the simulation environment.

31. (Original) The method of claim 1 wherein the first simulation model and the second simulation model are within different simulation domains in the simulation environment.

32. (Currently amended) A method comprising:

reading state information ~~form~~ from a first simulation model in a simulation environment when a simulation domain of the first simulation model is deactivated; and

writing the state information to a second simulation model in the simulation environment prior to activation of a simulation domain of the second simulation model, said first simulation model and said second simulation model representing different versions of a same functionality in a circuit design being simulated.

33. (Original) The method of claim 32 wherein the first simulation model and the second simulation model each have a particular level of performance and resolution, and wherein simulation of the circuit design switches from the first simulation model to the second simulation model is based on a change in a performance level and/or a resolution level desired at a different stage of simulation.

34. (Original) The method of claim 32 wherein the first simulation model and the second simulation model are among a plurality of simulation models representing a same functionality in the circuit design, each of the plurality of simulation models having a particular level of performance and resolution, and each of the plurality of simulation models being used at different stages of simulation depending on a desired performance level and/or resolution level of the simulation.

35. (Original) A machine readable medium having stored thereon machine executable instructions that when executed implement a method comprising:

identifying state information comprising a transfer from a first simulation model in a simulation environment, said transfer being directed to a second simulation model in a circuit design being simulated in the simulation environment;

receiving the state information from the first simulation model; and

making the state information available to the second simulation model without simulating the transfer in the circuit design.

36. (Original) A machine readable medium having stored thereon machine executable instructions that when executed implement a method comprising:

reading state information from a first simulation model in a simulation environment when a simulation domain of the first simulation model is deactivated; and

writing the state information to a second simulation model in the simulation environment prior to activation of a simulation domain of the second simulation model, said first simulation model and said second simulation model representing different versions of a same functionality in a circuit design being simulated.